ELSEVIER

Discrete Applied Mathematics 58 (1995) 157–168

DISCRETE APPLIED MATHEMATICS

New results on the completion time variance minimization☆

Wieslaw Kubiak

Faculty of Business Administration, Memorial University of Newfoundland, St. John's, Nfld., Canada A1B 3X5

Received 15 July 1992; revised 16 December 1992

Abstract

A single machine scheduling problem where the objective is to minimize the variance of job completion times is considered. The model is applicable to many environments where it is desirable to provide jobs, customers or computer files, with approximately the same service. It is shown that the problem can be formulated as a problem of maximizing a zero–one quadratic function which is a submodular function with a special cost structure. It immediately follows from the cost structure that value of the function for a sequence that minimizes total absolute deviation about a common unrestrictive due date is at most 100% smaller than the one for an optimal sequence for the completion time variance problem. The Monge property holds for the costs. Other simple properties of the function are also presented. A pair of dynamic programs for maximizing the function is proposed. The worst case time complexity of the best of the pair is $O(n^2\mathrm{spt})$, where spt is the mean flow time of an SPT-schedule for all jobs except the longest one.

Keywords: Completion time variance; Submodular functions; Monge property; Dynamic programming

1. Introduction

Consider a single machine with independent jobs all available for processing at time zero. The problem is to schedule the jobs nonpreemptively in such a way that the variance of job completion times is minimized. Merten and Muller [14] have been the first to consider the problem. They motivate the variance performance measure by computer file organization problems in which it is important to provide uniform response time to users. In the same spirit, Kanet [12] motivates the measure as being applicable to any service and manufacturing setting where it is desirable to provide jobs or customers with approximately the same service. Since the completion time variance (CTV) problem is equivalent to the problem of minimizing total squared

☆This research is supported by NSERC Grant OGP0105675.

SSDI 0166-218X(93)E0125-I

deviation about a common unrestrictive due date (TSD problem) [17, 2, 3], another motivation comes from just-in-time production where an ideal schedule would be one in which all jobs finish exactly on their assigned due dates. Although linear penalties for earliness and tardiness simplify algorithms minimizing total deviation, large deviations from the due date might be highly undesirable, which justifies nonlinear penalties and squared deviations in particular [3]. Vani and Raghavachari [17] formulate the CTV as a problem in statistics.

The completion time variance is a member of a broad family of equivalent measures of variation which includes the waiting time variance (WTV) [14, 9] and the total sum of squared deviation of completion times (TSDC) [12]. It is not our intention to present all the properties of the optimal schedules which have been so far obtained for these measures because they are comprehensively reviewed in Baker and Scudder [3] and succinctly summarized in Bagchi [1], Vani and Raghavachari [17], and Hall and Kubiak [10]. We restrict ourselves to the following essential facts:

(a) Any optimal schedule is V-shaped, i.e., the jobs, if any, prior to the shortest job are scheduled in nonincreasing order of their processing times, while the jobs, if any, after the shortest job are scheduled in nondecreasing order of their processing times [9].

(b) No optimal schedule includes idle time.

(c) Any optimal schedule starts with the longest job [16].

Based on these properties, a family of heuristics for the CTV problem is presented by Eilon and Chowdhury [9], Kanet [12], and Vani and Raghavachari [17]. Also, these properties are essential in the enumerative approaches of Bagchi et al. [2] (see also [6, 8]). When the measure of variation is total absolute difference of completion times the problem is shown by Kanet [12] to be solvable in $O(n \log n)$ time, where n is the number of jobs. Bagchi [1] considers bicriterion scheduling problems which involve both the CTV measure of variation and mean flow time.

De et al. [7], and independently Kahlbacher [11], show pseudopolynomial time algorithms for the CTV problem thus proving that it cannot be unary NP-hard. De et al. [7] report their pseudopolynomial algorithm to run in $O(n^2 \mathrm{Ms})$ time, where Ms is the makespan of all jobs. Cai [5] establishes the binary NP-completeness of the weighted CTV problem. In this paper, we show that the CTV problem can be formulated as a problem of maximizing a zero–one quadratic function which is a submodular function with a special cost structure. It immediately follows from the structure that value of the function for a sequence that minimizes total absolute deviation about a common unrestrictive due date is at most 100% smaller than the one for an optimal sequence for the completion time variance problem. The Monge property holds for the costs. A pair of dynamic programs for maximizing the function is also proposed. The worst case time complexity of the best of the pair is $O(n^2 \mathrm{spt})$, where spt is the mean flow time of an SPT-schedule of all jobs except the longest one. It has recently been shown that the problem of maximizing the functions is NP-hard [13], which proves that the CTV problem itself is NP-hard.

The outline of the paper is as follows: in Section 2, notation is introduced and the CTV problem is defined. In Section 3, the CTV problem is formulated as a quadratic (0, 1) program. In Section 4, it is shown that the program is equivalent to a problem of maximizing a submodular function. In Section 5, we present a pair of dynamic programs for maximizing the function. Some concluding remarks are given in Section 6.

2. Notation and the problem definition

Let $J = \{1, \ldots, n, n + 1\}$ be a set of jobs. Associated with each job $i \in J$ is an integer processing time $P_i > 0$, where jobs are numbered such that $P_1 \leqslant P_2 \leqslant \cdots \leqslant P_n \leqslant P_{n+1}$. We distinguish the longest job $n + 1$ from the remaining jobs in J since property (c) fixes it in the time interval $[0, P_{n+1}]$ in an optimal schedule for the CTV problem. Thus, in fact, we have to schedule n jobs from $I = J \setminus \{n + 1\}$. Moreover by fixing job $n + 1$ in $[0, P_{n+1}]$ we make our further considerations easier and our notation more convenient. This will be especially seen in the proofs of Lemmas 3 and 4 in the next section. Properties (b) and (c) enable us to represent any nonpreemptive schedule by a sequence of the jobs in I. Moreover, without loss of generality, we can assume that the sequence is not empty, i.e. $n \geqslant 1$. We define:

s – a sequence of the jobs in I. To simplify our notation we take $s(0) = n + 1$, i.e., job $n + 1$ appears in position 0.

i^* – the position of the shortest job in s, i.e., $s(i^*) = 1$.

V – the set of all V-shaped sequences of the jobs in I, i.e.,

$$V = \{s\colon P_{s(i)} \geqslant P_{s(i+1)} \text{ for } 1 \leqslant i < i^* \text{ and } P_{s(i)} \leqslant P_{s(i+1)} \text{ for } i^* \leqslant i < n\},$$

$C_{s(i)} = \sum_{0 \leqslant j \leqslant i} P_{s(j)}$ – the completion time of the job in position i in s,

$\mathrm{cost}(s) = \sum_{0 \leqslant j \leqslant n} (C_{s(j)} - C_{s(i^*)})^2$ – the cost of s measured with respect to the completion time of the shortest job,

$\Delta(s) = \sum_{0 \leqslant j \leqslant n} (C_{s(j)} - C_{s(i^*)})$ – the deviation imbalance of s with respect to the completion time of the shortest job,

$C_s = (\sum_{0 \leqslant j \leqslant n} C_{s(j)})/(n + 1)$ – the mean flow time of s,

$\mathrm{ctv}(s) = \sum_{0 \leqslant j \leqslant n} (C_{s(j)} - C_s)^2$ – the completion time variance of s,

$e_s = C_{s(i^*)} - P_{n+1}$ – the maximum early deviation from $C_{s(i^*)}$ in s,

$l_s = C_{s(n)} - C_{s(i^*)}$ – the maximum late deviation from $C_{s(i^*)}$ in s.

Owing to the previously mentioned properties (a) and (b), the CTV problem can be formulated as follows:

$$\text{(CTV)} \qquad \min_{s \in V} \{\mathrm{ctv}(s)\}.$$

3. A quadratic (0, 1) program for CTV

In this section, our purpose is to show that the CTV problem can be formulated as a quadratic (0, 1) program with a special cost structure. We begin with Lemma 1 which relates the completion time variance ctv(s) with cost(s) and $\Delta(s)$ for a sequence $s \in V$. Cost(s) represents total squared deviation of s from the completion time of the shortest job (i.e., job 1). Obviously, the completion time of job 1 may not coincide with the mean flow time of s. This is the case when the total early deviation from the completion time of job 1 in s differs from the total late deviation from the completion time of job 1 in s [10]. Then, the time difference which exists between the mean flow time of s and the completion time of job 1 is equal to $|\Delta(s)|/(n+1)$. Thus by subtracting $\Delta^2(s)/(n+1)$ from cost(s), we obtain the completion time variance of s.

Lemma 1. *For any sequence $s \in V$,*

$$\mathrm{ctv}(s) = \mathrm{cost}(s) - \Delta^2(s)/(n+1). \tag{1}$$

Proof. First,

$$C_s = \sum_{0 \leqslant j \leqslant n} C_{s(j)}/(n+1) = \left[\sum_{0 \leqslant j \leqslant n} (C_{s(j)} - C_{s(i^*)}) + (n+1)C_{s(i^*)}\right] \Big/ (n+1)$$
$$= C_{s(i^*)} + \Delta(s)/(n+1). \tag{2}$$

Then, from (2)

$$\mathrm{ctv}(s) = \sum_{0 \leqslant j \leqslant n} (C_{s(j)} - C_s)^2 = \sum_{0 \leqslant j \leqslant n} (C_{s(j)} - C_{s(i^*)} - \Delta(s)/(n+1))^2$$
$$= \mathrm{cost}(s) - 2\Delta(s) \sum_{0 \leqslant j \leqslant n} (C_{s(j)} - C_{s(i^*)})/(n+1) + \Delta^2(s)/(n+1)$$
$$= \mathrm{cost}(s) - 2\Delta^2(s)/(n+1) + \Delta^2(s)/(n+1)$$
$$= \mathrm{cost}(s) - \Delta^2(s)/(n+1)$$

as claimed in (1). □

To formulate a quadratic program for CTV, we replace the set V of all V-shaped sequences of the jobs in I by a set of vectors in $\{-1, 1\}^n$, where $x \in \{-1, 1\}^n$ corresponding to s is defined as follows:

$$x_i = \begin{cases} -1 & \text{if job } i \text{ appears before job 1 in } s, \\ 1 & \text{otherwise,} \end{cases} \tag{3}$$

for $i = 1, \ldots, n$. We assume that $x_1 = -1$. Lemmas 2, 3, 4 and 5 construct the quadratic program. Lemma 2 shows how to express the maximum early and late deviations from $C_{s(i^*)}$ in $s \in V$, by a vector in the new space.

Lemma 2. *For $s \in V$, $l_s = h + \alpha_s$ and $e_s = h - \alpha_s$, where $h = (\sum_{1 \leqslant i \leqslant n} P_i)/2$ and $\alpha_s = (\sum_{1 \leqslant i \leqslant n} x_i P_i)/2$.*

Proof. From the definitions of x_i, l_s and e_s we have

$$\sum_{1 \leqslant i \leqslant n} x_i P_i = l_s - e_s. \tag{4}$$

Moreover,

$$l_s + e_s = \sum_{1 \leqslant i \leqslant n} P_i. \tag{5}$$

From (4) and (5), we get $\sum_{1 \leqslant i \leqslant n} x_i P_i + \sum_{1 \leqslant i \leqslant n} P_i = 2l_s$, or

$$l_s = h + \alpha_s. \tag{6}$$

Finally, from (5) and (6) we get $e_s = h - \alpha_s$. □

Lemma 3. *For* $s \in V$,

$$\mathrm{cost}(s) = \sum_{1 \leqslant i \leqslant n} [a_i^2 + (n-i)P_i^2/4] + \sum_{1 \leqslant j < i \leqslant n} [(n-i)P_i P_j/2 + a_i P_j] x_i x_j, \tag{7}$$

where $a_i = P_i + (\sum_{1 \leqslant j \leqslant i-1} P_j)/2$, *for* $i = 1, \ldots, n$.

Proof. For $n > 1$, let t be a sequence s without job n. In s, job n may appear either immediately after job $n+1$ and therefore before job 1 or at the end of the sequence and therefore after job 1. In the former case, cost(s) exceeds cost(t) by r^2, where $r = e_t + P_n$ and e_t is maximum early deviation in t. In the latter case cost(s) exceeds cost(t) by q^2, where $q = l_t + P_n$ and l_t is maximum late deviation in t. From Lemma 2, we get $r = a_n - \alpha_t$ and $q = a_n + \alpha_t$. Thus, we have

$$\mathrm{cost}(s) = \begin{cases} \mathrm{cost}(t) + (a_n + x_n \alpha_t)^2 & \text{if } n \geqslant 2, \\ P_1^2 & \text{if } n = 1. \end{cases}$$

It is worth noticing that the symmetry of r and q is due to the assumption that job $n+1$ is scheduled in position 0. Since $x_n^2 = 1$, we get

$$\mathrm{cost}(s) = \begin{cases} \mathrm{cost}(t) + (a_n^2 + 2a_n x_n \alpha_t + \alpha_t^2) & \text{if } n \geqslant 2, \\ P_1^2 & \text{if } n = 1. \end{cases}$$

An induction on n gives (7) and completes the proof. □

Lemma 4. *For* $s \in V$,

$$\Delta^2(s) = \sum_{1 \leqslant i \leqslant n} [(n-i)P_i/2 + a_i]^2$$

$$+ \sum_{1 \leqslant j < i \leqslant n} 2[(n-i)P_i/2 + a_i][(n-j)P_j/2 + a_j] x_i x_j, \tag{8}$$

where a_i *is defined as in Lemma* 3, *for* $i = 1, \ldots, n$.

Proof. For $n > 1$, let t be sequence s without job n. In s, job n may appear either immediately after job $n+1$ and therefore before job 1 or at the end of the sequence and therefore after job 1. In the former case, imbalance $\Delta(s)$ changes by $r = -(e_t + P_n)$, where e_t is maximum early deviation in t. In the latter case, imbalance $\Delta(s)$ changes by $q = l_t + P_n$, where l_t is maximum late deviation in t. From Lemma 2 we get $r = \alpha_t - a_n$ and $q = \alpha_t + a_n$. Thus, we have

$$\Delta(s) = \begin{cases} \Delta(t) + \alpha_t + x_n a_n & \text{if } n \geqslant 2, \\ x_1 P_1 & \text{if } n = 1. \end{cases}$$

By induction on n we get $\Delta(s) = \sum_{1 \leqslant i \leqslant n} [(n-i)P_i/2 + a_i]x_i$ and $\Delta^2(s)$ as claimed in (8). $\square$

Lemma 5 summarizes the results obtained thus far giving a quadratic program for the CTV problem.

Lemma 5. *For $n \geqslant 2$, problem CTV is equivalent to program*

$$\text{(Q)} \qquad \min_{x \in \{-1,1\}^n} \left\{ \sum_{1 \leqslant j < i \leqslant n} d_{ij}/2\, x_i x_j \right\},$$

where $d_{ij} = \beta_i \gamma_j$ and

$$\beta_i = (n - \mathrm{i})P_i + 2a_i, \quad \gamma_j = (j+1)P_j - 2a_j, \qquad \textit{for } 1 \leqslant j < i \leqslant n.$$

Moreover, $d_{ij} \geqslant 0$, $1 \leqslant j < i \leqslant n$, and $\gamma_1 = 0$.

Proof. The first part follows immediately from Lemmas 1, 3, and 4. Since $P_1 \leqslant P_2 \leqslant \cdots \leqslant P_n$, we note that $(j+1)P_j \geqslant 2P_j + (\sum_{1 \leqslant k \leqslant j-1} P_k) = 2a_j$. Finally, $\gamma_1 = 0$ since $a_1 = P_1$. $\square$

Because $\gamma_1 = 0$, program (Q) does not depend on x_1. We assume $x_1 = -1$. Moreover, for $n = 2$, program (Q) assumes value 0 regardless of whether $x_2 = -1$ or $x_2 = 1$. Thus, both $s = 3, 1, 2$ and $s = 3, 2, 1$ are optimal for three jobs. Since optimal sequences for $n = 0, 1$ are trivial to obtain, we focus our attention on the case of $n \geqslant 3$.

So far in Lemmas 2, 3 and 4, we have enjoyed the symmetry given by our choice of values that the variables in (Q) are allowed to assume. The symmetry is also reflected in the objective function of (Q) which, roughly speaking, consists of a negative part (the variables in a product are of different signs – corresponding jobs lie on opposite sides of job 1) and a positive one (the variables in a product are of the same sign – corresponding jobs lie on the same side of job 1). Now, we change the space of solution from $\{-1,1\}^n$ to $\{0,1\}^n$ by the transformation $y = (x+e)/2$, where e is a unit n-dimensional vector.

Lemma 6. *For $n \geqslant 3$, program* (Q) *is equivalent to program*

$$\text{(P)} \qquad \max_{y \in \{0,1\}^n} \left\{ \sum_{2 \leqslant j < i \leqslant n} d_{ij}\, y_i \oplus y_j \right\},$$

where $y_i \oplus y_j = \bar{y}_i y_j + y_i \bar{y}_j$ and $\bar{y}_i = 1 - y_i$, for $i = 1, \ldots, n$.

Proof. By replacing x_i by $2y_i - 1$, for $i = 1, \ldots, n$, in (Q) we get $\sum_{1 \leqslant j < i \leqslant n} d_{ij}/2\, x_i x_j = \sum_{1 \leqslant j < i \leqslant n} d_{ij}/2 - \sum_{1 \leqslant j < i \leqslant n} d_{ij}\, y_i \oplus y_j$. Thus, since $d_{i,1} = 0$, for $i = 2, \ldots, n$, programs (P) and (Q) are equivalent. □

We make the following two observations on program (P).

Observation 1.

$$\beta_i \leqslant \beta_{i+1}, \quad \text{for } 1 \leqslant i < n.$$

Observation 2.

$$\gamma_i \leqslant \gamma_{i+1}, \quad \text{for } 1 \leqslant i < n.$$

Either observation may be used to prove optimality gap of at most 100% for an alternating solution to (P). A solution to (P) is said to be alternating iff it assigns 0 to all variables with even indices and 1 to all variables with odd indices, or the other way round. It is worth noticing that the alternating solutions minimize mean absolute deviation about a common unrestrictive due date (see [3]).

Theorem 1.

$$\text{Opt/Alt} \leqslant 2,$$

where Opt *and* Alt *are values of* (P) *for an optimal and an alternating solution respectively.*

Proof. From Observation 2, we have $\sum_{2 \leqslant j < i} d_{ij}/2 \leqslant \sum_{2 \leqslant j < i} d_{ij}\, y_i \oplus y_j$, for each i, $3 \leqslant i \leqslant n$, and any alternating solution. Thus, the theorem holds. □

Observations 1 and 2 also prove that the Monge property holds for d_{ij}'s,

Theorem 2. *Numbers d_{ij}, for $2 \leqslant j < i \leqslant n$, satisfy the following inequality:*

$$d_{i'j} + d_{ij'} \leqslant d_{ij} + d_{i'j'}, \quad \text{for } 3 \leqslant i < i' \leqslant n \quad \text{and} \quad 2 \leqslant j < j' \leqslant n - 1.$$

Proof. Follows immediately from Observations 1 and 2. □

4. CTV as a submodular function maximization

We note that the objective function of (P), i.e., function

$$f(y) = \sum_{2 \leqslant j < i \leqslant n} d_{ij} y_i \oplus y_j$$

is an integer-valued function on the subsets of $N = \{2, \ldots, n\}$, where the subsets are represented by their characteristic vectors from $\{0,1\}^{n-1}$. Moreover, the following Lemma 7 holds.

Lemma 7. *Function f is submodular.*

Proof. In what follows, we shall use the subsets of N as arguments of f rather than their characteristic vectors. Let T be a subset of N. Define nonnegative c_T and r_T for $T \subseteq \{2, \ldots, n\}$ as follows:

$$c_T = r_T = \begin{cases} d_{\max\{i,j\}, \min\{i,j\}} & \text{if } T = \{i,j\}, \\ 0 & \text{otherwise.} \end{cases}$$

For a subset S of N, we have

$$\begin{aligned} f(S) &= \sum_{i \in S, j \in N \setminus S, j < i} d_{ij} + \sum_{i \in N \setminus S, j \in S, j < i} d_{ij} \\ &= \sum_{T = \{i,j\}, T \cap S \neq \emptyset, T \cap (N \setminus S) \neq \emptyset} d_{\max\{i,j\}, \min\{i,j\}}, \\ &= \sum_{T \cap S \neq \emptyset, T \nsubseteq S} r_T. \end{aligned}$$

Therefore,

$$\sum_{T \cap S \neq \emptyset} r_T = f(S) + \sum_{T \subseteq S} c_T \quad \text{or} \quad f(S) = -\sum_{T \subseteq S} c_T + \sum_{T \cap S \neq \emptyset} r_T.$$

Thus by Proposition 7.1 of [15, p. 695] the lemma holds. □

Lemma 7 gives an interesting insight into the CTV problem although it does not provide us with any polynomial-time algorithm for this problem. This is due to the fact that the problem of maximizing a submodular function is NP-hard. Contrary to the problem of maximizing f, its minimization is trivial since $f(\emptyset) = f(N) = \min_{T \subseteq N} \{f(T)\} = 0$. Moreover, the following Lemma 8 follows immediately from the definition of f.

Lemma 8. *For $T \subseteq N, f(T) = f(N/T)$.*

Therefore, from Lemma 8 we observe that function f is not nondecreasing.

5. Dynamic programs for CTV

In this section we exploit the fact that $d_{ij} = \beta_i \gamma_j$, for $2 \leqslant j < i \leqslant n$, to obtain two dynamic programs maximizing f. In one of them the state is a function of β_i's and γ_j's are used as weights; in the other the state is a function of γ_j's and β_i's are used as weights. We call these two programs an r–c (rows or columns) pair of dynamic programs for CTV. We also provide simple upper bounds on the number of states of the programs. Thus, the program with the smaller bound can be selected for calculations. Let us begin with program for the rows. Define function

$$h_k(y) = \sum_{2 \leqslant j < i \leqslant n \text{ and } k \leqslant i} d_{ij} y_i \oplus y_j, \tag{9}$$

for $k = 3, \ldots, n$, and

$$h(k, \gamma) = \max_{y \in J(k, \gamma)} \{h_k(y)\}, \tag{10}$$

where $J(k,\gamma) = \{y \in \{0,1\}^n \colon \sum_{2 \leqslant j \leqslant k-1} \gamma_j y_j = \gamma\}$. Thus, $J(k,\gamma)$ is a set of all those schedules in which the sum of γ_j's of the jobs from $\{2, \ldots, k-1\}$ scheduled after the shortest job equals γ. To simplify notation in this section we take y in $\{0,1\}^n$ though only vectors with $y_1 = 0$ are being discussed. Define $\gamma_{k-1}^* = \sum_{2 \leqslant j \leqslant k-1} \gamma_j$ and $\gamma_{n-1}^* = \gamma^*$. From (9) and (10) we have

$$\begin{aligned} h_k(y) &= h_{k+1}(y) + \sum_{2 \leqslant j < k} d_{kj} y_k \oplus y_j \\ &= h_{k+1}(y) + \beta_k \bar{y}_k \sum_{2 \leqslant j \leqslant k-1} \gamma_j y_j + \beta_k y_k \left(\gamma_{k-1}^* - \sum_{2 \leqslant j \leqslant k-1} \gamma_j y_j \right), \end{aligned} \tag{11}$$

for $k = 3, \ldots, n$. By definition, $h_{n+1}(y) = 0$ for any $y \in \{0,1\}^n$. From (10) and (11) we get the following recurrence relation:

$$h(k,\gamma) = \max\{h(k+1,\gamma) + \beta_k \gamma, h(k+1, \gamma + \gamma_k) + \beta_k(\gamma_{k-1}^* - \gamma)\}, \tag{12}$$

where the former alternative corresponds to $y_k = 0$ and the latter corresponds to $y_k = 1$. Furthermore, from (9), (10), and Lemma 8 we have

$$\max_{y \in \{0,1\}^n} \{f(y)\} = \max_{y \in \{0,1\}^n} \{h_3(y)\} = \max\{h(3,0), h(3,\gamma_2)\} = h(3,0).$$

It is worth noticing that by Lemma 8 we may consider solutions with $y_2 = 0$ only. The recurrence (12) may be solved for $h(3,0)$ using the knowledge

$$h(n+1, \gamma) = 0 \quad \text{for all } \gamma. \tag{13}$$

The solution can be found in time $O(n\gamma^*)$. In order to estimate γ^* we prove the following lemma.

Lemma 9.

$$\gamma^* = \sum_{1 \leqslant j \leqslant n-1} (2j - n) P_j.$$

Proof. Induction on n. $\square$

Thus, $\gamma^* = n(2\text{rspt} - \text{Ms} - P_n)$, where rspt is the mean flow time of a reversed SPT-schedule of the jobs in I, and MS is the makespan of the jobs in I. We now give program for the columns. Define function

$$g_k(y) = \sum_{2 \leqslant j < i \leqslant n \text{ and } j \leqslant k} d_{ij} y_i \oplus y_j, \tag{14}$$

for $k = 2, \dots, n-1$, and

$$g(k, \beta) = \max_{y \in I(k,\beta)} \{g_k(y)\}, \tag{15}$$

where $I(k,\beta) = \{y \in \{0,1\}^n\colon \sum_{k+1 \leqslant j \leqslant n} \beta_j y_j = \beta\}$. Thus, $I(k,\beta)$ is a set of all those schedules in which the sum of β_j's of the jobs from $\{k+1, \dots, n\}$ scheduled after the shortest job equals β. Define $\beta_k^* = \sum_{k+1 \leqslant j \leqslant n} \beta_j$ and $\beta_2^* = \beta^*$. From (14) and (15) we have

$$\begin{aligned} g_k(y) &= g_{k-1}(y) + \sum_{k < i \leqslant n} d_{ik}\, y_i \oplus y_k \\ &= g_{k-1}(y) + \gamma_k \bar{y}_k \sum_{k < i \leqslant n} \beta_i y_i + \gamma_k y_k \Bigl(\beta_k^* - \sum_{k < i \leqslant n} \beta_i y_i\Bigr), \end{aligned} \tag{16}$$

for $k = 2, \dots, n-1$. By definition, $g_1(y) = 0$ for any $y \in \{0,1\}^n$. From (15) and (16) we get the following recurrence relation:

$$g(k,\beta) = \max\{g(k-1,\beta) + \beta\gamma_k, g(k-1, \beta + \beta_k) + \gamma_k(\beta_k^* - \beta)\}, \tag{17}$$

where the former alternative corresponds to $y_k = 0$ and the latter corresponds to $y_k = 1$. Furthermore, from (14), (15), and Lemma 8 we have

$$\max_{y \in \{0,1\}^n} \{f(y)\} = \max_{y \in \{0,1\}^n} \{g_{n-1}(y)\} = \max\{g(n-1, 0), g(n-1, \beta_n)\} = g(n-1, 0).$$

By Lemma 8 we may restrict ourselves to solutions with $y_n = 0$ only. The recurrence (17) may be solved for $g(n-1, 0)$ using the knowledge

$$g(1,\beta) = 0 \quad \text{for all } \beta. \tag{18}$$

The solution can be found in time $O(n\beta^*)$. In order to estimate β^*, we prove the following lemma.

Lemma 10.

$$\beta^* = \sum_{3 \leqslant i \leqslant n} 2(n - i + 1)P_i + (n - 2)(P_1 + P_2).$$

Proof. Induction on n. □

It is worth noticing that $\beta^* = 2n\,\mathrm{spt} - (n + 2)P_1 - nP_2$ where spt is the mean flow time of an SPT-schedule of the jobs in I.

Example (Kanet [12]). Let $n = 6$ and $P_1 = 2$, $P_2 = 3$, $P_3 = 6$, $P_4 = 9$, $P_5 = 21$, $P_6 = 65$. Then $\gamma^* = 88$ and $\beta^* = 336$. We calculate:

$$\gamma_2 = 1, \quad \beta_3 = 35,$$

$$\gamma_3 = 7, \quad \beta_4 = 47,$$

$$\gamma_4 = 16, \quad \beta_5 = 83,$$

$$\gamma_5 = 64, \quad \beta_6 = 171.$$

Because $\gamma^* < \beta^*$, we solve (12) using (13), and obtain the following optimal sequence: $h(3,0)$, $h(4,0)$, $h(5,0)$, $h(6,0) = 15\,048$ with the evaluation $y_6 = 1$, $y_5 = y_4 = y_3 = 0$. Thus, the optimal sequence obtained, i.e. 5, 4, 3, 2, 1, 6 is the same as in Kanet [12].

6. Concluding remarks

We conjecture that the results presented in this paper for the CTV problem can be extended to other scheduling problems with a common due date, e.g. the problem of minimizing mean absolute deviation. That means, we conjecture that this problem reduces to a problem of maximizing zero–one quadratic function ($\sum e_j y_j + \sum c_{ij} y_i y_j$) which is a submodular function with a special cost structure (i.e. $c_{ij} = a_i b_j$) and that an r–c pair of dynamic programs can be formulated for solving the latter. The conjecture, if proved, would provide a natural framework for many problems with a common due date.

The submodularity has not been exploited in the r–c pair of dynamic programs for solving the CTV problem, actually, the main fact we used was the special structure of d_{ij}. Nevertheless, given its theory (see for example [15]), the submodularity can be exploited in many different ways, e.g. in further developing and analyzing branch and bound as well as approximate algorithms for CTV.

In the CTV problem, the mean flow time of a schedule sets up a service standard for the system. Any job below or above the standard incurs a penalty. Obviously, generally, the service standard may be chosen differently, e.g. it may be set up equal to the completion of a distinguished job e.g. the shortest one. This gives rise to the problems in which it is a job completion time rather than a due date, which determines a system service standard. These problems avoid the question where the due date should be located with respect to the schedule (see factor $\Delta^2(s)/(n+1)$ in (1)). Thus, one can hope for more efficient algorithms for such problems. For example, one can readily observe from (7), that the best of the r–c pair of dynamic programs for cost(s) runs in time O(nMS).

Acknowledgements

The author is grateful to two anonymous referees for their helpful comments on an earlier draft of the paper.

References

[1] U. Bagchi, Simultaneous minimization of mean and variation of flow time and waiting time in single machine systems, Oper. Res. 37 (1989) 118–125.
[2] U. Bagchi, R. Sullivan and Y. Chang, Minimizing mean squared deviation of completion times about a common due date, Management Sci. 33 (1987) 894–906.
[3] K.R. Baker and G.D. Scudder, Sequencing with earliness and tardiness penalties: a review, Oper. Res. 38 (1990) 22–36.
[4] E.R. Barnes and A.J. Hoffman, Partitioning, spectra and linear programming, in: R.W. Pulleyblank, ed., Progress in Combinatorial Optimization (Academic Press, New York, 1984) 13–25.
[5] X. Cai, Variance minimization in single machine systems: complexity and algorithm, Research Report, Department of Mathematics, University of Western Australia (1991).
[6] P. De, J.B. Ghosh and C.E. Wells, A note on the minimization of mean squared deviation of completion times about a common due date, Management Sci. 35 (1989) 1143–1147.
[7] P. De, J.B. Ghosh and C.E. Wells, On the minimization of completion time variance with a bi-criteria extension, Oper. Res., to appear.
[8] P. De, J.B. Ghosh and C.E. Wells, Scheduling about a common due date with earliness and tardiness penalties, Comput. Res. 17 (1990) 231–241.
[9] S. Eilon and I.G. Chowdhury, Minimizing waiting time variance in the single machine problem, Management Sci. 23 (1977) 567–575.
[10] N.G. Hall and W. Kubiak, Proof of a conjecture of Schrage about the completion time variance problem, Oper. Res. Lett. 10 (1991) 567–575.
[11] H.G. Kahlbacher, SWEAT – A program for a scheduling problem with earliness and tardiness penalties, European J. Oper. Res. 43 (1989) 111–112.
[12] J.J. Kanet, Minimizing variation of flow time in single machine systems, Management Sci. 27 (1981) 1453–1459.
[13] W. Kubiak, Completion time variance minimization on single machine is difficult, Oper. Res. Lett. 14 (1993) 49–59.
[14] A.G. Merten and M.E. Muller, Variance minimization in single machine sequencing problems, Management Sci. 18 (1972) 518–528.
[15] G.L. Nemhauser and L.A. Wolsey, Integer and Combinatorial Optimization (Wiley, New York, 1988).
[16] L. Schrage, Minimizing the time-in-system variance for a finite jobset, Management Sci. 21 (1975) 540–543.
[17] V. Vani and M. Raghavachari, Deterministic and random single machine sequencing with variance minimization, Oper. Res. 35 (1987) 111–120.